

**TUGAS AKHIR - KI141502**

# **Rancang Bangun Pixel Art Converter Menggunakan Segmentasi berbasis K-means Clustering**

**YUNA SUGIANELA**  
5113100035

Dosen Pembimbing  
Dr.Eng. NANIK SUCIATI, S.Kom., M.Kom.  
MAULIDAN BAGUS R A, S.Kom.

JURUSAN TEKNIK INFORMATIKA  
Fakultas Teknologi Informasi  
Institut Teknologi Sepuluh Nopember  
Surabaya 2017

***[Halaman ini sengaja dikosongkan]***



**TUGAS AKHIR - KI141502**

# **Rancang Bangun Pixel Art Converter Menggunakan Segmentasi berbasis K-means Clustering**

**YUNA SUGIANELA**  
5113100035

Dosen Pembimbing I  
Dr.Eng. NANIK SUCIATI, S.Kom., M.Kom.

Dosen Pembimbing II  
MAULIDAN BAGUS A R, S.Kom.

JURUSAN TEKNIK INFORMATIKA  
Fakultas Teknologi Informasi  
Institut Teknologi Sepuluh Nopember  
Surabaya, 2017

***[Halaman ini sengaja dikosongkan]***



**FINAL PROJECT - KI141502**

# **Pixel Art Converter Using K-means Clustering Segmentation**

**YUNA SUGIANELA**  
5113100035

Supervisor I  
Dr.Eng. NANIK SUCIATI, S.Kom., M.Kom.

Supervisor II  
MAULIDAN BAGUS A R, S.Kom.

DEPARTMENT OF INFORMATICS  
FACULTY OF INFORMATION TECHNOLOGY  
Sepuluh Nopember Institute of Technology  
Surabaya, 2017

***[Halaman ini sengaja dikosongkan]***

## LEMBAR PENGESAHAN

### RANCANG BANGUN PIXEL ART CONVERTER MENGUNAKAN SEGMENTASI BERBASIS K-MEANS CLUSTERING

#### TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat  
Memperoleh Gelar Sarjana Komputer  
pada  
Rumpun Mata Kuliah Komputasi Cerdas dan Visi  
Program Studi S-1 Jurusan Teknik Informatika  
Fakultas Teknologi Informasi  
Institut Teknologi Sepuluh Nopember

Oleh:  
**YUNA SUGIANELA**  
**NRP: 5113 100 035**

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Dr.Eng. NANIK SUCIATI, S.Kom., M.Kom.  
NIP. 197104281994122001

MAULIDAN BAGUS A R, S.Kom.



**SURABAYA**  
**JUNI, 2017**

***[Halaman ini sengaja dikosongkan]***



# **Rancang Bangun Pixel Art Converter Menggunakan Segmentasi Berbasis K-means Clustering**

Nama Mahasiswa : Yuna Sugianela  
NRP : 5113 100 035  
Jurusan : Teknik Informatika, FTIf ITS  
Dosen Pembimbing 1 : Dr.Eng. Nanik Suciati, S.Kom., M.Kom.  
Dosen Pembimbing 2 : Maulidan Bagus A R, S.Kom.

## ***Abstrak***

*Pixel art merupakan jenis aset grafis yang digunakan pada game. Untuk mengefisiensi pekerjaan pada industri game, diperlukan sebuah converter citra raster biasa menjadi pixel art. Tahapan dalam membangun aplikasi pixel art converter adalah mengubah warna menjadi aturan yang berlaku pada pixel art, lalu membuat tepi gambar menjadi jaggy yang merupakan ciri khas grafis pixel art.*

*Algoritma segmentasi yang digunakan merupakan segmentasi berbasis k-means clustering. Segmentasi ini berguna untuk mengubah warna citra menjadi lebih sederhana. Citra hasil dari segmentasi k-means kemudian diolah menjadi citra yang memiliki tepian bergerigi atau jaggy yang merupakan ciri khas utama dari pixel art.*

*Aplikasi dinyatakan telah memenuhi kebutuhan Tim Desain Maulidan Games. Kualitas citra hasil pixel art converter dipengaruhi oleh ukuran citra input, parameter nilai K untuk k-means clustering serta skala grid untuk membuat tepian jaggy. Nilai optimal yang digunakan untuk membuat pixel art yang baik yaitu, untuk citra gradient color nilai optimal nilai K untuk K-means adalah 25, sedangkan untuk citra flat color menggunakan nilai K 16, nilai skala grid untuk membuat tepian adalah 80, ukuran citra optimal adalah 100 x 100 piksel.*

***Kata kunci: Pixel Art, Segmentasi, K-means, Jaggy***

***[Halaman ini sengaja dikosongkan]***

# Pixel Art Converter Using K-means Clustering Segmentation

Student Name : Yuna Sugianela  
Registration Number : 5113 100 035  
Department : Informatics Engineering, FTIf ITS  
First Supervisor : Dr. Eng. Nanik Suciati, S.Kom.,M.Kom  
Second Supervisor : Maulidan Bagus A R, S.Kom.

## *Abstract*

*Pixel art is a type of graphic assets used in games. For job efficiency in the gaming industry, an ordinary raster image converter is required to be pixel art. Stages in building pixel art converter application are change the color to the rules that apply to pixel art, then make the image edge to jaggy which is characteristic of pixel art graphics.*

*The segmentation algorithm used is k-means clustering based segmentation. Segmentation is useful to change the color of the image becomes more simple. The result image of k-means segmentation is then processed into an image that has jagged edge or jaggy which is the main characteristic of pixel art.*

*Applications fulfills the needs of Maulidan Games Design Team. Image quality of pixel art converter's result is influenced by input image size, K value parameter for k-means clustering and grid scale to create jaggy edge. The optimal values that are used to create a good pixel art are: for the image gradient color the optimal value of K for K-means is 25, whereas for flat color image using the value of K 16, the grid scale value to create the edge is 80, the optimal image size is 100 x 100 pixels.*

**Keywords:** *Pixel Art, Segmentation, K-means, Jaggy*

***[Halaman ini sengaja dikosongkan]***

## KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Segala puji bagi Allah SWT atas rahmat dan anugerah-Nya sehingga penulis dapat menyelesaikan tugas akhir berjudul Rancang Bangun *Pixel Art Converter* Menggunakan Segmentasi Berbasis *K-means Clustering*. Pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Ibu Dr. Eng. Nanik Suciati, S.Kom., M.Kom dan Mas Maulidan Bagus, S.Kom sebagai pembimbing dalam tugas akhir ini,
2. Segenap staf dan dosen pengajar Teknik Informatika ITS,
3. Mas Hildi Radya N dan segenap tim Maulidan Games yang telah membantu pengerjaan tugas akhir ini,
4. Orang tua, Ibu Suherwin dan Bapak Alm. Sugiyono, Nenek Hj. Basri dan seluruh keluarga sebagai motivasi utama penulis serta selalu memberikan doa dan limpahan kasih sayang kepada penulis,
5. Prasetya Gilang Nuswantara yang selalu memberikan bantuan, semangat, dan dukungan pada penulis selama dua jenjang pendidikan terakhir ini,
6. Kharisma Nur Anissa, Nida Amalia, Okta Farida Khairunnisa sebagai sahabat perjuangan di kontrakan, serta Tities Novaninda dan Mardiana Sekarsari sahabat sejak Mahasiswa Baru yang selalu memberikan semangat dan nasehat, serta sebagai tempat mencurahkan segala keluh kesah setiap hari,
7. Imla Amelia Ulinnuha, Mita Ayu Maratika, dan Jhevi Istansti sebagai sahabat yang meskipun jauh namun tetap memberikan semangat kepada penulis,
8. Admin Laboratorium KCV khususnya Rizky Haqiqi, Reza Andriyunanto, Ihsan Prasetya, Rizqi Okta, Nida Amalia, dan seluruh angkatan mulai 2011, 2012, 2014, dan 2015 yang selama tiga tahun terakhir perkuliahan selalu menjadi teman

main, teman serius, teman ambisius, dan selalu memberikan pembelajaran baru pada bidang KCV,

9. User TA KCV 2013 dan seluruh ‘sahabat KCV’ sebagai teman perjuangan dalam mengerjakan tugas-tugas kuliah khususnya tugas akhir ini,
10. Seluruh teman perjuangan di BEM ITS Berani dan Wahana Juang, Ukafo, ITS Expo, KMI, HMTC, dan BEM FTIf yang telah mewarnai dunia perkuliahan penulis di bidang non akademik, pengembangan diri, serta masa-masa pendewasaan diri penulis,
11. TC 2013 sebagai teman perjuangan sejak masuk di jurusan hingga sekarang,
12. Serta semua pihak yang tidak dapat disebutkan satu persatu yang telah banyak memberikan berbagai macam bantuan.

Penulis menyadari bahwa pengerjaan tugas akhir ini tidak luput dari kekurangan. Untuk itu dalam kesempatan ini penulis mohon maaf atas segala kesalahan. Kritik dan saran yang membangun dapat disampaikan melalui email [yuna.sugianela13@mhs.if.its.ac.id](mailto:yuna.sugianela13@mhs.if.its.ac.id). Semoga tugas akhir ini berguna bagi berbagai pihak.

Surabaya, 05 Juni 2017

Penulis

## DAFTAR ISI

<b>LEMBAR PENGESAHAN.....</b>	<b>Error! Bookmark not defined.</b>
<b><i>Abstrak</i>.....</b>	<b>vii</b>
<b><i>Abstract</i>.....</b>	<b>ix</b>
<b>KATA PENGANTAR.....</b>	<b>xi</b>
<b>DAFTAR ISI.....</b>	<b>xiii</b>
<b>DAFTAR GAMBAR.....</b>	<b>xv</b>
<b>DAFTAR TABEL .....</b>	<b>xvi</b>
<b>DAFTAR KODE SUMBER.....</b>	<b>xvii</b>
<b>BAB I PENDAHULUAN.....</b>	<b>1</b>
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah.....	2
1.3 Batasan Masalah .....	2
1.4 Tujuan Tugas Akhir .....	2
1.5 Manfaat Tugas Akhir .....	2
<b>BAB II DASAR TEORI.....</b>	<b>3</b>
2.1 <i>Pixel Art</i> .....	3
2.2 Konversi Citra yang Berbasis Pikel .....	4
2.3 Citra Digital .....	7
2.3.1 Ruang Warna .....	8
2.3.2 Jenis Citra Digital pada Maulidan Games .....	9
2.4 Segmentasi Citra .....	10
2.5 K-means Clustering .....	11
2.6 Jarak <i>Euclidean</i> .....	12
<b>BAB III ANALISIS DAN PERANCANGAN.....</b>	<b>15</b>
3.1 Tahap Analisis .....	15
3.1.1 Deskripsi Umum .....	15
3.1.2 Spesifikasi Kebutuhan Sistem .....	15
3.1.3 Analisis Permasalahan .....	16
3.2 Tahap Perancangan .....	17
3.2.1 Perancangan Sistem .....	17
3.2.2 Perancangan Data .....	17

3.2.3 Perancangan Proses.....	18
<b>BAB IV IMPLEMENTASI.....</b>	<b>23</b>
4.1 Lingkungan Implementasi.....	23
4.1.1 Perangkat Keras.....	23
4.2 Implementasi Tahap Segmentasi Berbasis <i>K-means</i> <i>Clustering</i> .....	24
4.3 Implementasi Pembuatan Tepian <i>Jaggy</i> .....	29
<b>BAB V UJI COBA DAN EVALUASI .....</b>	<b>33</b>
5.1 Lingkungan Uji Coba.....	33
5.2 Data Uji Coba.....	33
5.3 Skenario Uji Coba.....	36
5.4 Uji Coba dengan Variasi Ukuran Citra Input .....	36
5.5 Uji Coba dengan Variasi Nilai K untuk <i>K-means</i> <i>Clustering</i> .....	40
5.6 Uji Coba dengan Variasi Skala Grid untuk Membuat Tepian <i>Jaggy</i> .....	43
5.7 Uji Coba Kebergunaan.....	47
5.8 Evaluasi.....	50
<b>BAB VI KESIMPULAN DAN SARAN.....</b>	<b>53</b>
6.1 Kesimpulan.....	53
6.2 Saran .....	54
<b>LAMPIRAN.....</b>	<b>55</b>
<b>DAFTAR PUSTAKA.....</b>	<b>81</b>
<b>BIODATA PENULIS.....</b>	<b>83</b>



## DAFTAR GAMBAR

<b>Gambar 2. 1</b> Contoh grafis <i>pixel art</i> .....	4
<b>Gambar 2. 2</b> Konsep Konversi .....	4
<b>Gambar 2. 3</b> Input Grafis yang Akan Dikonversi .....	5
<b>Gambar 2. 4</b> Grafis Output .....	5
<b>Gambar 2. 5</b> Input Grafis .....	6
<b>Gambar 2. 6</b> Output Grafis .....	6
<b>Gambar 2. 7</b> Representasi citra digital .....	7
<b>Gambar 2. 8</b> Representasi citra digital.....	8
<b>Gambar 2. 9</b> Ruang warna RGB .....	9
<b>Gambar 2. 10</b> Citra <i>gradient color</i> .....	10
<b>Gambar 2. 11</b> Citra <i>flat color</i> .....	10
<b>Gambar 3. 1</b> Citra <i>pixelated</i> karena <i>gradient color</i> .....	16
<b>Gambar 3. 2</b> Diagram alir keseluruhan proses <i>Pixel Art Converter</i> .....	18
<b>Gambar 3. 3</b> Diagram alir metode klastering <i>k-means</i> .....	20
<b>Gambar 3. 4</b> Diagram alir pembuatan tepian <i>jaggy</i> .....	21
<b>Gambar 5. 1</b> Perbandingan citra dengan segmentasi dan tanpa segmentasi .....	51

## DAFTAR TABEL

<b>Tabel 5. 1</b>	Spesifikasi lingkungan uji coba .....	33
<b>Tabel 5. 2</b>	Tabel citra untuk uji coba.....	33
<b>Tabel 5. 3</b>	Tahap pengujian ukuran citra .....	37
<b>Tabel 5. 4</b>	Tabel hasil uji coba variasi ukuran input .....	38
<b>Tabel 5. 5</b>	Tahap perubahan citra pada pengujian nilai K .....	40
<b>Tabel 5. 6</b>	Tabel hasil uji coba variasi nilai K .....	42
<b>Tabel 5. 7</b>	Tahap pengujian skala grid tepian .....	44
<b>Tabel 5. 8</b>	Tabel hasil uji coba variasi nilai skala grid .....	45
<b>Tabel 5. 9</b>	Pertanyaan pengujian kebergunaan .....	47
<b>Tabel 5. 10</b>	Hasil kuesioner pengguna .....	49

## DAFTAR KODE SUMBER

<b>Kode Sumber 4. 1</b>	Pemanggilan kelas ‘KMeans’ .....	24
<b>Kode Sumber 4. 2</b>	Penetapan <i>Centroid</i> secara acak .....	25
<b>Kode Sumber 4. 3</b>	Iterasi perhitungan jarak dan penempatan data .....	26
<b>Kode Sumber 4. 4</b>	Perhitungan <i>centroid</i> baru.....	27
<b>Kode Sumber 4. 5</b>	Pengecekan nilai <i>centroid</i> baru dan lama ....	28
<b>Kode Sumber 4. 6</b>	Input nilai skala grid .....	29
<b>Kode Sumber 4. 7</b>	Menyimpan informasi piksel grid .....	30
<b>Kode Sumber 4. 8</b>	Penentuan warna dominan grid .....	31

***[Halaman ini sengaja dikosongkan]***

# **BAB I**

## **PENDAHULUAN**

Pada bab ini dibahas hal-hal yang mendasari tugas akhir. Bahasan meliputi latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika laporan tugas akhir.

### **1.1 Latar Belakang**

Salah satu industri *game* digital di Surabaya yang sedang berkembang adalah Maulidan Games. Saat ini Maulidan Games sedang mengerjakan *Framework Project* yang diberi nama Diamond. Diamond Development Engine memiliki kemampuan dari semua *engine internal*, mulai dari *speech recognition*, *framework* untuk *game* dengan tampilan isometris, manajemen target perhari, *template game*, hingga fitur yang memungkinkan *engine* memprogram *game* sendiri.

Salah satu fitur dari Diamond yang diharapkan adalah sebuah konverter grafis dua dimensi menjadi bentuk *pixel art*. Grafis yang dimaksud adalah aset-aset *game* berupa desain karakter, tombol-tombol, background dan lain sebagainya. *Pixel art* merupakan jenis desain yang digunakan pada *game* tahun 1990-an seperti Pacman, Mario Bros, Galaga, dan lain lain. *Pixel art* memiliki ciri khas bentuk yang kotak-kotak, tepian gambar yang *jaggy*, serta menggunakan palet warna 8 bit atau 16 bit saja. Beberapa desainer melihat hal tersebut sebagai nostalgia pada zaman keemasan konsol generasi ketiga. Mereka berpendapat akan mendapatkan grafis yang lebih estetik dalam *pixel art*.

Harapan dari dikembangkannya *Pixel Art Converter* pada *Framework Diamond* dapat membantu produktivitas dari Maulidan Games. Karena dengan adanya *pixel art converter* ini dapat memudahkan *designer* untuk membuat aset-aset *game* berupa *pixel art* dengan efisien. Grafis *pixel art* juga diharapkan mampu menghemat penggunaan data karena penggunaan palet warna yang lebih sederhana.

## 1.2 Rumusan Masalah

Rumusan masalah yang terdapat pada tugas akhir ini adalah:

1. Bagaimana proses konversi grafis menggunakan algoritma segmentasi berbasis *k-means clustering*?
2. Bagaimana evaluasi hasil konversi citra *pixel art*?

## 1.3 Batasan Masalah

Batasan masalah pada tugas akhir ini adalah:

1. Aplikasi dengan platform desktop ini dibangun dengan bahasa pemrograman C#. Aplikasi dibangun dengan IDE Visual Studio.
2. Input citra pada uji coba menggunakan ukuran panjang dan lebar yang sama.

## 1.4 Tujuan Tugas Akhir

Tujuan tugas akhir ini adalah sebagai membuat *Pixel Art Converter* pada *Framework* Diamond 4.0 menggunakan segmentasi berbasis *k-means clustering*.

## 1.5 Manfaat Tugas Akhir

Manfaat tugas akhir ini adalah antara lain:

1. Memudahkan kinerja *designer* Maulidan Games untuk membuat *assets game* dengan model *pixel art*.
2. Efisiensi waktu dan penyimpanan data untuk meningkatkan produktivitas Maulidan Games.

## **BAB II**

### **DASAR TEORI**

Pada bab ini akan membahas tentang teori-teori yang digunakan dalam mengerjakan tugas akhir ini. Dalam pengerjaan tugas akhir ini digunakan beberapa teori dalam tahap – tahap pengerjaannya. Teori – teori yang dijelaskan pada bab ini adalah tentang citra digital yang menjadi input utama pada tugas akhir ini, *pixel art* yang merupakan output pada tugas akhir ini, dan segmentasi berbasis *k-means clustering*.

#### **2.1 Pixel Art**

*Pixel art* merupakan jenis desain yang digunakan pada game tahun 1990-an seperti Pacman, Mario Bros, Galaga, dan lain lain. *Pixel art* adalah karya seni digital yang diciptakan melalui grafis *raster software* yang diedit pada *pixel level*. *Pixel art* pertama kali ditemukan oleh Adele Goldberg dan Robert Flegal dari Xerox Palo Alto Research Center pada tahun 1982. [1]

Menurut analisa yang dilakukan tim desain Maulidan Games, *pixel art* memiliki ciri khas bentuk kotak-kotak, tepian gambar yang *jaggy* atau bergerigi, serta menggunakan palet warna yang cenderung *vintage*. Pembuatan *pixel art* diawali dari pemberian warna pada tiap piksel, yang mana tiap piksel merepresentasikan satu warna. Jika grafis tersebut diperbesar untuk dapat dijadikan *asset game*, grafis tersebut harus tetap merepresentasikan bentuk grid yang memuat satu warna.

*Pixel art* kini dipandang sebagai reaksi terhadap industri grafis tiga dimensi oleh pembuat *game* amatir dan penggemar dunia grafis. Banyak *graphic designer* yang menyukai konsep retro sering memilih menggunakan gaya ini dalam karya grafisnya. Beberapa desainer melihat hal tersebut sebagai nostalgia pada zaman keemasan konsol generasi ketiga. Mereka berpendapat akan mendapatkan grafis yang lebih estetik dalam *pixel art*. [2]



(a)



(b)

**Gambar 2. 1** Contoh grafis *pixel art* (a) *Game Mario Bros* [3] (b) *Game Infectonator Survivors* [4]

## 2.2 Konversi Citra yang Berbasis Pikel

Saat ini sudah dibangun aplikasi yang mampu mengonversi grafis menjadi model *pixelated*. Contoh aplikasi tersebut adalah <http://www.pixel-stitch.net/> dan <http://www194.lunapic.com>. Kedua aplikasi ini memiliki kelebihan dan kekurangan masing-masing, berikut adalah penjelasannya,

1. <http://www.pixel-stitch.net/>

Aplikasi ini menerapkan konsep pola border kristik yang mirip dengan pola kotak-kotak piksel.



**Gambar 2. 2** Konsep Konversi [5]

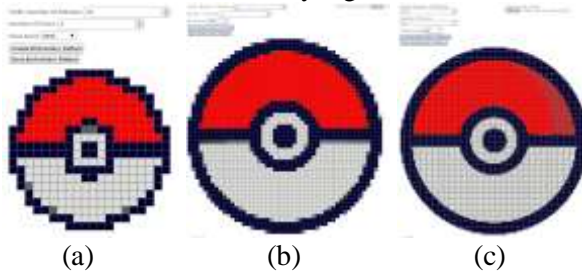
Pengguna dapat memberikan input berupa grafis dengan ekstensi file seperti .jpg dan .png. Berikut adalah contoh input yang diberikan pengguna.





**Gambar 2. 3** Input Grafis yang Akan Dikonversi [5]

Pengguna dapat mengubah *number of stitch* atau banyak kotak piksel yang diinginkan. Hal ini yang akan menentukan berapa ukuran grafis baru dan akan mempengaruhi pemetaan warna yang dilakukan pada grafis yang dihasilkan. Selain itu pengguna dapat mengubah banyak variasi warna yang digunakan. Berikut adalah contoh konversi yang dilakukan,



**Gambar 2. 4** Grafis Output (a) *Number of Stitches*: 20 dan *Number of Colors*: 5 (b) *Number of Stitches*: 50 dan *Number of Colors*: 10 (c) *Number of Stitches*: 100 dan *Number of Colors*: 20 [5]

Kelebihan dari aplikasi ini adalah kemampuannya meng-kustom input dari pengguna dari segi warna dan ukuran. Satu variasi input, yaitu dengan ukuran piksel 20 x 20 dan variasi warna 5, dapat memberikan ciri khas *pixel art* yang baik, yaitu kotak-kotak dan pinggiran *jaggy* yang rapi serta satu grid piksel

terisi satu macam warna. Namun, beberapa output lain memberikan hasil yang tidak memenuhi ciri khas *pixel art* serta output yang diberikan hanya dapat diunduh dengan ekstensi file .pdf. Sehingga tidak dapat dimanfaatkan untuk aset grafis sebuah *game*.

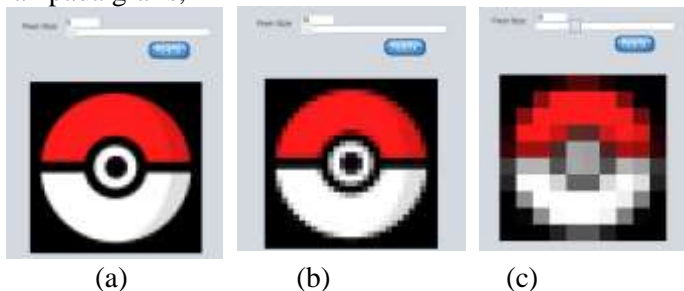
## 2. <http://www194.lunapic.com>

Aplikasi ini memberikan berbagai fitur konversi gambar secara online, salah satunya adalah *Pixelate*. Pada aplikasi ini, pengguna dapat memberikan variasi input berupa skala perubahan ukuran grafis. Berikut merupakan contoh input yang diberikan,



**Gambar 2. 5** Input Grafis [6]

Berikut contoh variasi input perubahan ukuran yang diberikan pada grafis,



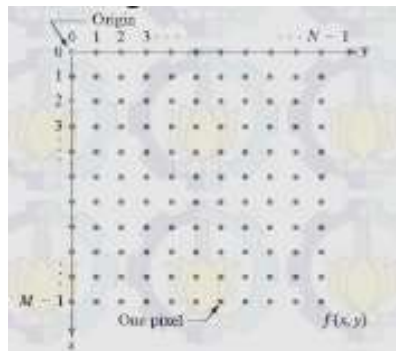
**Gambar 2. 6** Output Grafis (a) Skala Perubahan: 5 (b) Skala Perubahan: 10 (c) Skala Perubahan: 30 [6]

Hasil dari aplikasi ini kurang memenuhi ciri khas dari *pixel art*. Efek *jaggy* pada tepian grafis kurang sempurna, karena terdapat *shadow* berwarna abu-abu yang mengitarinya serta dalam grid piksel tidak hanya merepresentasikan satu jenis warna. Namun, untuk grafis hasil yang masih diinginkan dapat diunduh dalam ekstensi .png untuk dapat dimanfaatkan.

### 2.3 Citra Digital

Citra adalah gambar pada bidang dua dimensi. Ditinjau dari sudut pandang matematis, citra merupakan fungsi kontinu dari intensitas cahaya pada bidang dua dimensi.

Citra dibentuk dari persegi empat yang teratur sehingga jarak horizontal dan vertikal antara piksel satu dengan yang lain adalah sama pada seluruh bagian citra. Indeks  $x$  bergerak ke bawah dan indeks  $y$  bergerak ke kanan. Untuk menunjukkan koordinat digunakan posisi kanan bawah dalam citra berukuran  $m \times n$  piksel. Gambar berikut menunjukkan koordinat pada suatu citra digital. [7]



**Gambar 2. 7** Representasi citra digital [7]

Citra digital dapat direpresentasikan dalam bentuk matriks. Misalkan citra berukuran  $M \times N$  ( $M$  baris dan  $N$  kolom), maka representasi citranya ditunjukkan pada matriks berikut ini:

$$\begin{pmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{pmatrix}$$

**Gambar 2. 8** Representasi citra digital

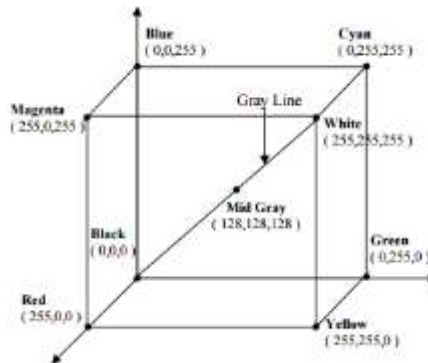
### 2.3.1 Ruang Warna

Warna pada dasarnya merupakan hasil persepsi dari cahaya dalam spektrum wilayah yang terlihat oleh retina mata, dan memiliki panjang gelombang antara 400nm sampai dengan 700nm.

Ruang warna atau yang sering juga disebut sebagai model warna merupakan sebuah cara atau metode untuk mengatur, membuat dan memvisualisasikan warna. Untuk aplikasi yang berbeda ruang warna yang dipakai bisa juga berbeda, hal ini dikarenakan beberapa peralatan tertentu memang membatasi secara ketat ukuran dan jenis ruang warna yang dapat digunakan. Ruang warna biasa digunakan untuk menganalisis citra. Beberapa ruang warna tersebut adalah RGB (*Red Green Blue*), HSL (*Hue Saturation Lightness*), HSV (*Hue Saturation Value*), HSI (*Hue Saturation Intensity*), dan HCL (*Hue Chroma Lightness*) dan YUV, YDbDr, YIQ dan YCbCr (*Luminance – Chrominance*) [8].

#### 2.3.1.1 *Red-Green-Blue* (RGB)

Sistem ruang warna RGB merupakan sistem ruang warna dasar. diperkenalkan oleh *National Television System Committee* (NTSC) yang banyak digunakan untuk menampilkan citra berwarna pada monitor CRT. Sistem ini diilustrasikan menggunakan sistem koordinat tiga dimensi seperti gambar berikut.



**Gambar 2. 9** Ruang warna RGB [9]

Pada gambar di atas tampak bahwa setiap warna akan diwakili oleh tiga buah nilai dalam koordinat tersebut yang menyatakan komponen warna RGB, sebagai misal warna merah akan diwakili oleh titik  $(255,0,0)$ . Rentang nilai untuk setiap sumbu berkisar dari 0 sampai 255. Pada gambar tersebut tampak juga bahwa warna cyan, magenta dan kuning merupakan komplemen warna merah, hijau, dan biru. [9]

### 2.3.2 Jenis Citra Digital pada Maulidan Games

Pada industri *game* Maulidan Games, terdapat dua jenis desain grafis yang digunakan untuk membangun aset-aset karakter *game*. Jenis desain yang pertama adalah citra dengan *gradient color* dan yang kedua adalah *flat color*.

Warna gradien dibuat dengan menggunakan dua atau lebih warna yang berbeda untuk melukis satu elemen sambil perlahan memudar di antara keduanya. Gradien memungkinkan desainer menciptakan sesuatu yang terasa seperti warna baru. Nada yang tidak ada sebelumnya, sesuatu yang terlihat unik, modern dan menyenangkan. [10]



**Gambar 2. 10** Citra *gradient color* [11]

Prinsip dasar dari desain *flat* yaitu layar komputer mewakili lingkungan digital dua dimensi yang bahkan benda tiga dimensi dari dunia nyata. Elemen antarmuka pengguna disederhanakan: bentuk grafis abstrak digunakan dan spasi diisi dengan warna yang tebal. [12]



**Gambar 2. 11** Citra *flat color* [13]

## 2.4 Segmentasi Citra

Segmentasi citra merupakan salah satu bagian penting dari pemrosesan citra, yang bertujuan untuk membagi citra menjadi beberapa region yang homogen berdasarkan kriteria kemiripan tertentu. Segmentasi citra berwarna sangat bermanfaat dalam berbagai aplikasi. Dari hasil segmentasi, identifikasi region-region dan objek-objek dalam citra dapat dilakukan, yang mana

sangat bermanfaat untuk analisis citra atau anotasi. Ada beberapa algoritma yang dapat digunakan untuk segmentasi citra berwarna, antara lain pendekatan berbasis model stockhastic, region growing berbasis morphological watershed, energy diffusion, graph partitioning, serta metode evaluasi kuantitatif. Namun, karena sulitnya masalah segmentasi, hanya sedikit algoritma yang dapat bekerja dengan baik pada data bervariasi yang sangat besar. [14]

Teknik pemetaan warna merupakan bagian salah satu teknik segmentasi citra menggunakan metode klusterisasi. Hal ini disebabkan dalam memetakan warna dari citra masukan akan dikelompokkan sesuai dengan kesamaan-kesamaan warna yang dimiliki. Sehingga tahap-tahap yang akan digunakan mempunyai kesamaan dengan metode klusterisasi. [15]

*Clustering* (klusterisasi) merupakan salah satu metode yang diterapkan secara luas dalam segmentasi citra dan statistik. Konsep utama *clustering* adalah menggunakan *centroid* untuk mewakili setiap *cluster* dan berdasarkan pada kesamaan dengan *centroid cluster* untuk dikelompokkan. Menurut karakteristik algoritma *clustering*, kita dapat secara kasar membagi ke dalam *clustering* "hirarkis" dan "partitional". [16]

## 2.5 K-means Clustering

Algoritma K-Means merupakan metode *non-heirarchial* yang pada awalnya mengambil sebagian dari banyaknya komponen dari populasi untuk dijadikan pusat *cluster* awal. Pada tahap ini pusat *cluster* dipilih secara acak dari sekumpulan populasi data. Berikutnya *K-means* menguji masing-masing komponen di dalam populasi data dan menandai komponen tersebut ke salah satu pusat *cluster* yang telah didefinisikan tergantung dari jarak minimum antar komponen dengan tiap-tiap pusat *cluster*. Posisi pusat *cluster* akan dihitung kembali sampai semua komponen data digolongkan ke dalam tiap-tiap pusat *cluster* dan terakhir akan terbentuk posisi pusat *cluster* baru. Beberapa alternatif penerapan *K-means* dengan beberapa

pengembangan teori-teori penghitungan terkait telah diusulkan. Secara umum algoritma dasar dari *K-means clustering* adalah sebagai berikut [17]:

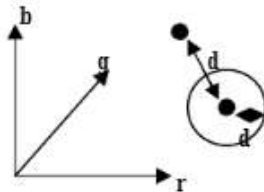
1. Tentukan jumlah *cluster*
2. Alokasikan data ke dalam *cluster* secara random
3. Hitung *centroid*/rata-rata dari data yang ada di masing-masing *cluster*
4. Alokasikan masing-masing data ke *centroid*/rata-rata terdekat
5. Kembali ke Step 3, apabila masih ada data yang berpindah *cluster* atau apabila perubahan nilai *centroid*, ada yang di atas nilai *threshold* yang ditentukan atau apabila perubahan nilai pada *objective function* yang digunakan di atas nilai *threshold* yang ditentukan.

## 2.6 Jarak Euclidean

Secara umum jarak *Euclidean* adalah jarak antara dua titik yang akan diukur pada satu, dua atau tiga dimensi. Berikut adalah posisi dari dua titik pada 3 dimensi:  $P = (p_x, p_y, p_z)$ ,  $Q = (q_x, q_y, q_z)$ , sehingga jaraknya:

$$r = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2 + (p_z - q_z)^2} \quad (2.1)$$

dengan P adalah nilai data dan Q adalah pusat dari kelompok data. Jika diterapkan dalam segmentasi citra menggunakan pemetaan warna dapat dijabarkan sebagai berikut:



$$d = \|(R, G, B) - (r_c, g_c, b_c)\| \quad (2.2)$$

$$d < d\theta \quad (2.3)$$



dengan  $d$  adalah jarak *Euclidean*.

Suatu contoh adalah ketika mencari jarak yang minimum antara dua titik di permukaan ruang 3 dimensi. Satu cara untuk memulainya dengan membuat suatu titik koordinat di atas permukaan masing-masing, dan bandingkan jarak dari tiap-tiap titik koordinat pada permukaan yang pertama dengan tiap-tiap titik koordinat pada permukaan yang kedua.

Penggolongan jarak yang minimum digunakan untuk menggolongkan data citra yang tak dikenal ke dalam kelompok-kelompok yang memperkecil jarak antara data citra dan kelompok di dalam jarak yang bervariasi. [15]

*[Halaman ini sengaja dikosongkan]*

## **BAB III**

### **ANALISIS DAN PERANCANGAN**

Bab ini membahas tentang analisis dan perancangan yang dilakukan untuk mempersiapkan pengerjaan tugas akhir. Tahap analisis meliputi analisis data, masalah, serta kebutuhan sistem. Selanjutnya pada tahap perancangan meliputi perancangan sistem, data, serta proses.

#### **3.1 Tahap Analisis**

Tahap analisis mendefinisikan kebutuhan yang akan dipenuhi dalam pembangunan aplikasi *Pixel Art Converter*. Selain itu dijelaskan pula alasan pengerjaan masing-masing pada tugas akhir ini.

##### **3.1.1 Deskripsi Umum**

Pada tugas akhir ini dibangun aplikasi *Pixel Art Converter* menggunakan algoritma klasterisasi *K-means*. Data masukan yang digunakan adalah citra asset karakter dari Maulidan Games. Data keluaran dari aplikasi ini adalah citra yang telah diproses menjadi *pixel art*.

Aplikasi ini diharapkan dapat membantu Tim Maulidan Games dalam produksi *game* digital khususnya dalam bagian desain grafis dalam menghasilkan citra *pixel art* yang di-*generate* secara otomatis dari citra asset biasa dengan cepat.

##### **3.1.2 Spesifikasi Kebutuhan Sistem**

Pada aplikasi *Pixel Art Converter* dalam tugas akhir ini, dibutuhkan proses yang dapat memenuhi kebutuhan sistem. Proses tersebut antara lain:

1. *K-means clustering*

*K-means clustering* digunakan untuk segmentasi citra supaya menghasilkan citra baru dengan jumlah warna yang lebih sederhana.

## 2. Pembuatan tepian *jaggy*

Pembuatan tepian ini ditujukan untuk membuat citra menyerupai bentuk khas dari *pixel art*.

### 3.1.3 Analisis Permasalahan

Citra *pixel art* hasil dari aplikasi *Pixel Art Converter* akan digunakan oleh tim desainer Maulidan Games. Untuk mendapatkan hasil citra *pixel art* yang baik terdapat permasalahan yang perlu dibahas, yaitu:

#### 3.1.3.1 Analisis Permasalahan *K-means Clustering*

Terdapat dua jenis citra yang dibuat oleh desainer grafis tim Maulidan Games seperti yang dibahas pada sub-bab 2.3.2. Citra dengan *gradient color* akan memberikan hasil yang *pixelated*, tepian yang tidak rapi, sehingga hasil tidak dapat digunakan oleh tim Maulidan Games.



**Gambar 3. 1** Citra *pixelated* karena *gradient color* [6]

Penyederhanaan diperlukan supaya menghasilkan warna yang lebih sederhana, sehingga didapatkan hasil *pixel art* yang lebih baik.

### 3.1.3.2 Analisis Permasalahan Pembuatan Tepian Jaggy

Tepian *jaggy* merupakan ciri khas utama dari citra *pixel art* yang membuat citra terlihat kotak-kotak seperti titik-titik piksel namun bukan citra yang *pixelated*. Bentuk *jaggy* ini dibuat dengan membuat grid pada citra. Grid pada citra dibuat dengan membagi citra menjadi kelompok-kelompok piksel. Jumlah piksel dalam grid sesuai dengan skala. Setelah piksel terkelompok dalam grid, masing-masing grid akan dicari warna dominan untuk mewakili warna pada grid tersebut.

Masalah yang terjadi adalah tidak adanya nilai optimal untuk skala grid. Oleh karena itu, akan diberikan kebebasan input skala untuk tim Maulidan Games.

## 3.2 Tahap Perancangan

Tahap perancangan dilakukan untuk merancang proses secara keseluruhan berdasarkan fungsionalitas dan kebutuhan dari aplikasi *Pixel Art Converter*.

### 3.2.1 Perancangan Sistem

Perancangan sistem dilakukan untuk menggambarkan proses secara keseluruhan dari *Pixel Art Converter*. Proses ini dimulai dengan memasukkan data citra asset karakter dari Maulidan Games. Kemudian dilakukan proses Segmentasi yang menggunakan algoritma dari *k-means clustering*. Citra asset karakter ini akan mengalami tahap *k-means clustering* untuk mendapatkan region-region warna pada citra yang ditentukan sesuai klaster *k-means*. Langkah terakhir adalah menjadikan tepian citra menjadi bentuk bergerigi atau *jaggy* yang merupakan ciri khas utama dari citra *pixel art*.

### 3.2.2 Perancangan Data

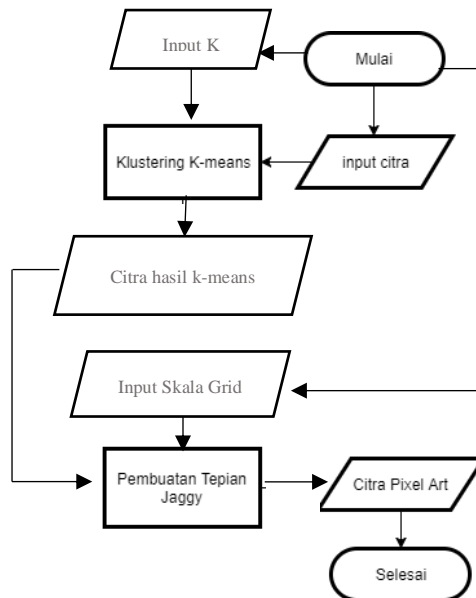
Data yang digunakan untuk implementasi perangkat lunak ini dibagi menjadi dua bagian utama, yaitu data masukan serta data keluaran.

Data masukan adalah 10 file citra karakter dari Maulidan Games yaitu berupa file citra dengan format .jpg atau .png. Data masukan selanjutnya adalah nilai  $K$  yang digunakan sebagai parameter jumlah kluster warna yang diinginkan pengguna. Yang terakhir adalah skala grid untuk membuat tepian *jaggy*.

Data keluaran terakhir adalah citra hasil klasterisasi *k-means*, dan citra hasil akhir dengan tepian *jaggy* yang merupakan hasil utama dari aplikasi ini.

### 3.2.3 Perancangan Proses

Perancangan proses dilakukan untuk memberikan gambaran mengenai setiap proses yang terdapat pada sistem *pixel art converter* lebih detail. Ada beberapa proses yang harus dilakukan yaitu klustering *K-means* dan pembuatan tepian *jaggy*.



**Gambar 3. 2** Diagram alir keseluruhan proses *Pixel Art Converter*

### 3.2.3.1 K-means Clustering

Segmentasi pada citra dapat dihasilkan dari proses pengelompokan warna-warna piksel menggunakan metode *K-means*. Tahap pertama dari klustering *k-means* adalah memilih *K* buah titik *centroid* secara acak. Kemudian pengelompokkan data sehingga terbentuk *K* buah kluster dengan titik *centroid* dari setiap kluster merupakan titik *centroid* yang telah dipilih sebelumnya. Proses pengelompokkan data ke dalam suatu cluster dapat dilakukan dengan cara menghitung jarak terdekat dari suatu data ke sebuah titik centroid. Perhitungan jarak Minkowski dapat digunakan untuk menghitung jarak antar 2 buah data. Rumus untuk menghitung jarak tersebut adalah:

$$d(x_i, x_j) = \left( |x_{i1} - x_{j1}|^g + |x_{i2} - x_{j2}|^g + |x_{i3} - x_{j3}|^g \right)^{\frac{1}{g}} \quad (3.1)$$

dimana:

$g = 1$ , untuk menghitung jarak Manhattan

$g = 2$ , untuk menghitung jarak Euclidean

$g = \infty$ , untuk menghitung jarak Chebychev

$x_i, x_j$  adalah dua buah data yang akan dihitung jaraknya

$p$  = dimensi dari sebuah data

Perbaharui nilai titik *centroid*. Pembaruan nilai *centroid* dapat dilakukan dengan rumus sebagai berikut:

$$\mu_k = \frac{1}{N_k} \sum_{q=1}^{N_k} q_k \quad (3.2)$$

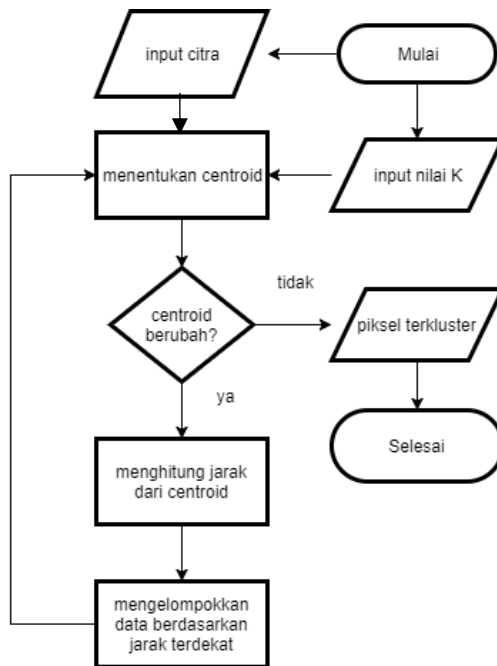
dimana:

$\mu_k$  = titik centroid dari *cluster* ke-*K*

$N_k$  = banyaknya data pada *cluster* ke-*K*

$x_q$  = data ke-*q* pada *cluster* ke-*K*

Selanjutnya, ulangi langkah 2 dan 3 sampai nilai dari titik *centroid* tidak lagi berubah.

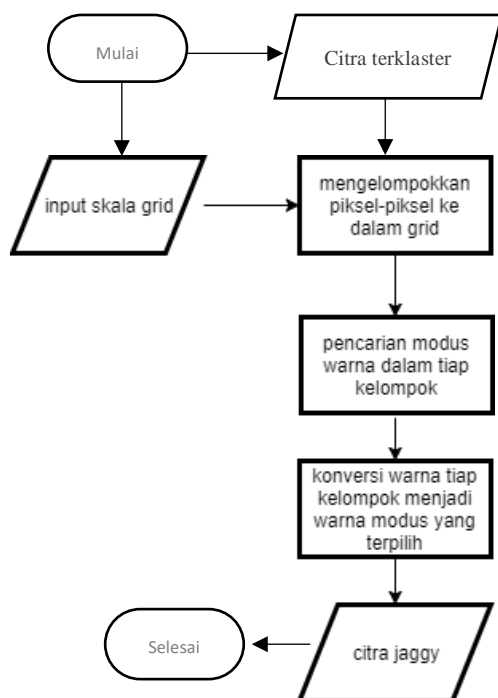


**Gambar 3. 3** Diagram alir metode klastering *k-means*

### 3.2.3.2 Pembuatan Tepian Jaggy

Tepian bergerigi merupakan ciri khas dari citra *pixel art*. Diperlukan variasi input skala grid yang digunakan untuk proses ini. Tahap pertama dari pembuatan tepian *jaggy* ini dimulai dengan mengelompokkan piksel-piksel citra ke dalam skala yang telah diinputkan. Kemudian tiap kelompok piksel citra ditentukan warna yang dominan. Warna dominan tersebut digunakan untuk mengubah seluruh warna dalam kelompok piksel. Cara ini dapat mengubah bentuk citra memiliki tepian yang terkotak-kotak atau *jaggy*.





**Gambar 3. 4** Diagram alir pembuatan tepian *jaggy*

***[Halaman ini sengaja dikosongkan]***

## **BAB IV**

### **IMPLEMENTASI**

Pada bab ini diuraikan mengenai implementasi perangkat lunak dari rancangan metode yang telah dibahas pada Bab III meliputi kode program dalam perangkat lunak. Selain itu, implementasi dari tiap proses, parameter masukan, keluaran, dan beberapa keterangan yang berhubungan dengan program juga dijelaskan.

#### **4.1 Lingkungan Implementasi**

Pada bagian ini akan dijelaskan tentang perangkat yang digunakan dalam melakukan pengujian. Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam uji coba seperti pada tabel berikut.

##### **4.1.1 Perangkat Keras**

Objek citra yang akan diolah pada implementasi tugas akhir ini adalah citra asset karakter dari Maulidan Games yang berformat .png atau .jpg.

Lingkungan implementasi pada tugas akhir ini menggunakan sebuah *personal computer* (PC). Detail perangkat dan spesifikasi dapat dilihat pada tabel di bawah ini.

**Table 4. 1** Spesifikasi perangkat yang digunakan

<b>Perangkat</b>	<b>Spesifikasi</b>
Merk dan Tipe	TOSHIBA C40
Prosesor	Core i5-5200U @ 2.20GHz
RAM	4 GB
HDD	500 GB
OS	Microsoft Windows 10
Software	Microsoft Visual Studio 2015

## 4.2 Implementasi Tahap Segmentasi Berbasis *K-means Clustering*

Pada tahap segmentasi berbasis *k-means clustering* diperlukan input citra karakter serta parameter nilai K yang diinginkan. Proses *k-means clustering* dilakukan pada kelas ‘KMeans2’, parameter yang dibutuhkan adalah citra input, jumlah K, dan ruang warna yang diinginkan. **Kode Sumber 4.1** berikut menunjukkan cara pemanggilan kelas KMeans2 yang dinyatakan dalam variabel `_kMeans`. Variabel `b` merupakan citra dengan tipe data bitmap, `trackbarcek` merupakan input nilai K yang diinputkan pengguna melalui *toolbox trackbar* pada Visual Studio.

```
1. _kMeans = new KMeans2(b, trackbarcek, ImageProcessor.C
  olour.Types.RGB);
```

### Kode Sumber 4. 1 Pemanggilan kelas ‘KMeans’

Tahap pada kelas ‘KMeans2’ dimulai dengan menetapkan nilai *centroid* secara acak dalam *clustering*. Fungsi pada **Kode Sumber 4.2** berikut menunjukkan proses penetapan *centroid* tersebut.

```
1. public KMeans2(Bitmap bmp, int numCluster, Colour.Type
  s model)
2.     {
3.         _image = (Bitmap)bmp.Clone();
4.         _processedImage = (Bitmap)bmp.Clone();
5.         _model = model;
6.
7.         _previousCluster = new Dictionary<string,
  Cluster>();
8.         _currentCluster = new Dictionary<string, C
  luster>();
9.         FindTopXColours(numCluster); //find top X
  colours in the image
10.        //create clusters for top X colours
```

```

11.         for (int i = 0; i < _topColours.Length; i+
12.             +)
13.             {
14.                 PixelData pd = Colour.GetPixelData(_to
15.                 pColours[i].R, _topColours[i].G, _topColours[i].B, mod
16.                 el);
17.                 _previousCluster.Add(_topColours[i].Na
18.                 me, new Cluster(pd.Ch1, pd.Ch2, pd.Ch3));
19.                 _currentCluster.Add(_topColours[i].Nam
20.                 e, new Cluster(pd.Ch1, pd.Ch2, pd.Ch3));
21.             }
22.         }

```

#### Kode Sumber 4. 2 Penetapan *Centroid* secara acak

Tahap selanjutnya adalah iterasi untuk semua data, dilakukan perhitungan jarak antara *centroid* dari *cluster* dan data. Kemudian pindahkan data ke kelompok *cluster* dengan jarak yang lebih kecil. Perhitungan jarak menggunakan jarak Euclidean yang dilakukan pada data piksel citra dengan ruang warna RGB. **Kode Sumber 4.3** berikut merupakan implementasi perhitungan tersebut.

```

1. private void AllocateToCluster(PixelData pd)
2.     {
3.         //find distance of this colour from each c
4.         luster centroid
5.         Dictionary<string, Distance> distances = n
6.         ew Dictionary<string, Distance>();
7.         foreach (KeyValuePair<string, Cluster> c i
8.         n _currentCluster)
9.         {
10.             float d = (float)Math.Sqrt(
11.                 (double)Math.Pow((c.Value.Centroid
12.                 R - pd.Ch1), 2) +
13.                 (double)Math.Pow((c.Value.Centroid
14.                 G - pd.Ch2), 2) +

```

```

11.                (double)Math.Pow((c.Value.Centroid
    B - pd.Ch3), 2)
12.                );
13.                distances.Add(c.Key, new Distance(d));
14.            }
15.
16.            //allocate this colour to the closest cluster based on distance
17.            List<KeyValuePair<string, Distance>> list
    = new List<KeyValuePair<string, Distance>>();
18.            list.AddRange(distances);
19.
20.            list.Sort(delegate (KeyValuePair<string, Distance> kvp1, KeyValuePair<string, Distance> kvp2)
21.            { return Comparer<float>.Default.Compare(kvp1.Value.Measure, kvp2.Value.Measure); });
22.
23.            //assign to closest cluster
24.            if (_pixelDataClusterAllocation.ContainsKey(list[0].Key))
25.            {
26.                ((List<PixelData>) _pixelDataClusterAllocation[list[0].Key]).Add(pd);
27.            }
28.            else
29.            {
30.                List<PixelData> clrList = new List<PixelData>();
31.                clrList.Add(pd);
32.                _pixelDataClusterAllocation.Add(list[0].Key, clrList);
33.            }
34.        }

```

**Kode Sumber 4. 3** Iterasi perhitungan jarak dan penempatan data

Tahap selanjutnya adalah melakukan perhitungan *centroid* baru dari *cluster-cluster* baru yang terbentuk.

```

1. private void CalculateClusterCentroids()
2.     {
3.         foreach (KeyValuePair<string, Cluster> cluster in _currentCluster)
4.         {
5.             List<PixelData> clrList = (List<PixelData>)_pixelDataClusterAllocation[cluster.Key];
6.             float cR = 0;
7.             float cG = 0;
8.             float cB = 0;
9.             foreach (PixelData clr in clrList)
10.            {
11.                cR += clr.Ch1;
12.                cG += clr.Ch2;
13.                cB += clr.Ch3;
14.
15.                if (!_clusterColours.ContainsKey(clr.Name))
16.                {
17.                    _clusterColours.Add(clr.Name, Color.FromArgb((int)cluster.Value.CentroidR, (int)cluster.Value.CentroidG, (int)cluster.Value.CentroidB));
18.                }
19.            }
20.            float count = clrList.Count + 1; //total of colours plus 1 for the existing centroid
21.            cluster.Value.CentroidR = (cluster.Value.CentroidR + cR) / count; //average to find new centroid
22.            cluster.Value.CentroidG = (cluster.Value.CentroidG + cG) / count;
23.            cluster.Value.CentroidB = (cluster.Value.CentroidB + cB) / count;
24.        }
25.    }

```

#### Kode Sumber 4. 4 Perhitungan *centroid* baru

Langkah terakhir adalah dilakukan pengecekan terhadap *centroid* baru, apakah memiliki nilai yang sama dengan *centroid* lama. Apabila hasil *centroid* baru sama, maka iterasi dihentikan.

**Kode Sumber 4.5** berikut merupakan implementasi dari pengecekan nilai *centroid* tersebut.

```

1. private void CheckConvergence()
2.     {
3.         //if current and previous cluster centroids
         are the same then converged
4.         bool match = true;
5.         foreach (KeyValuePair<string, Cluster> cluster in _currentCluster)
6.         {
7.             if (((int)cluster.Value.CentroidR != (int)_previousCluster[cluster.Key].CentroidR)
8.                 && ((int)cluster.Value.CentroidG != (int)_previousCluster[cluster.Key].CentroidG)
9.                 && ((int)cluster.Value.CentroidB != (int)_previousCluster[cluster.Key].CentroidB))
10.            {
11.                match = false;
12.                break;
13.            }
14.        }
15.        if (!match)
16.        {
17.            foreach (KeyValuePair<string, Cluster> cluster in _currentCluster)
18.            {
19.                _previousCluster[cluster.Key].CentroidR = cluster.Value.CentroidR;
20.                _previousCluster[cluster.Key].CentroidG = cluster.Value.CentroidG;
21.                _previousCluster[cluster.Key].CentroidB = cluster.Value.CentroidB;
22.            }
23.        }
24.        _converged = match;
25.    }

```

**Kode Sumber 4. 5** Pengecekan nilai *centroid* baru dan lama



### 4.3 Implementasi Pembuatan Tepian *Jaggy*

Pada tahap pembuatan tepian *jaggy*, diperlukan input skala ukuran grid oleh pengguna. Skala grid digunakan untuk membuat grid pada citra input. Skala grid berasal dari input pengguna yang disimpan dalam variabel ‘trackbarcek’.

```

1. int trackbarcek = trackBar1.Value;
2. double pembagi = Math.Round((double)(width / trackb
   arcek));
3. Color[][] x = new Color[(width/ (int)pembagi)*(width
   h/ (int)pembagi)][];
4. for (int i = 0; i < (width / (int)pembagi) * (width
   / (int)pembagi); i++)
5. {
6.     x[i] = new Color[(int)(pembagi * pembagi)];
7. }

```

#### Kode Sumber 4. 6 Input nilai skala grid

Informasi piksel pada setiap grid disimpan pada array 2 dimensi yang dilambangkan dengan variabel x seperti yang terlihat pada baris 13 **Kode Sumber 4.7** di bawah ini.

```

1. //mencari warna terbanyak
2. int id = 0;
3. int ij = 0;
4. for (int i = 0; i <= width-
   pembagi; i=i+(int)pembagi)
5. {
6.     for(int j = 0; j <= height-
   pembagi; j=j+ (int)pembagi)
7.     {
8.         id = 0;
9.         for (int ix = i; ix < (i + (int)pembagi)
   ; ix++)
10.        {
11.            for (int jx = j; jx < (j + (int)pembagi); j
   x++)

```

```

12.     {
13.         x[ij][id] = b.GetPixel(ix, jx);
14.         id++;
15.     }
16. }
17. ij++;
18. }
19. }

```

#### Kode Sumber 4. 7 Menyimpan informasi piksel grid

Selanjutnya dalam setiap grid dilakukan pencarian warna dominan, yaitu dengan cara menentukan frekuensi setiap warna yang ada pada setiap grid. Kemudian frekuensi tersebut diurutkan secara *descending*, urutan tersebut menyatakan warna dominan yang terpilih. Warna dominan yang terpilih disimpan pada array mode seperti yang ditunjukkan pada baris 5 **Kode Sumber 4.8** di bawah ini.

```

1. for (int i = 0; i < width -
    pembagi; i += (int)pembagi)
2. {
3.     for (int j = 0; j < height -
        pembagi; j += (int)pembagi)
4.     {
5.         mode[ij2] = x[ij2].GroupBy(v => v).OrderByDescen
            ding(g => g.Count()).First().Key;
6.         id2 = 0;
7.         for (int ix = i; ix < i + (int)pembagi; ix++)
8.         {
9.             for (int jx = j; jx < j + (int)pembagi; jx+
                +)
10.            {
11.                Color xxx = Color.FromArgb(255, mode[ij2].R
                    , mode[ij2].G, mode[ij2].B);
12.                b.SetPixel(ix, jx, xxx);
13.                id2++;
14.            }
15.        }

```

```
16.     ij2++;  
17. }  
18. }
```

#### **Kode Sumber 4. 8** Penentuan warna dominan grid

Pada **Kode Sumber 4.8** juga dilakukan implementasi penggantian warna dalam grid dengan warna dominan yang terpilih. Hal tersebut ditunjukkan pada baris7 hingga 18.

***[Halaman ini sengaja dikosongkan]***

## BAB V

### UJI COBA DAN EVALUASI

Dalam bab ini dibahas mengenai hasil uji coba sistem yang telah dirancang dan dibuat. Uji coba dilakukan untuk mengetahui kinerja sistem dengan lingkungan uji coba yang telah ditentukan.

#### 5.1 Lingkungan Uji Coba

Pada bagian ini akan dijelaskan tentang perangkat yang digunakan dalam melakukan pengujian. Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam uji coba seperti pada tabel berikut.


**Tabel 5. 1** Spesifikasi lingkungan uji coba

<b>Perangkat Keras</b>	Prosesor: Intel Core i5
	Memori: 4 GB
<b>Perangkat Lunak</b>	Sistem Operasi: Ms Windows 10
	Perangkat Lunak Pembangun: Ms. Visual Studio





#### 5.2 Data Uji Coba


Citra yang digunakan dalam aplikasi ini berformat .jpg atau .png. Ukuran citra memiliki lebar dan panjang yang sama). Untuk menguji aplikasi *pixel art converter* diperlukan 10 citra asset karakter *game* pada Maulidan Games dengan variasi ukuran dan jenis yang berbeda. Berikut data citra yang digunakan:

**Tabel 5. 2** Tabel citra untuk uji coba

No	Citra	Variasi Ukuran (piksel x piksel)			Jenis Citra
1		90	270	512	<i>Gradient</i>

No	Citra	Variasi Ukuran (piksel x piksel)			Jenis Citra
2		90	270	512	<i>Gradient</i>
3		90	270	512	<i>Flat</i>
4		90	270	512	<i>Flat</i>
5		90	270	512	<i>Flat</i>

No	Citra	Variasi Ukuran (piksel x piksel)			Jenis Citra
6		90	270	512	<i>Gradient</i>
7		90	270	512	<i>Flat</i>
8		90	270	512	<i>Gradient</i>
9		90	270	512	<i>Flat</i>

No	Citra	Variasi Ukuran (piksel x piksel)			Jenis Citra
		90	270	512	
10					<i>Gradient</i>

### 5.3 Skenario Uji Coba

Uji coba dilakukan untuk mengetahui nilai-nilai parameter yang tepat untuk digunakan pada masing-masing proses. Nilai parameter yang tepat penting untuk diketahui karena penggunaan parameter yang tepat akan memberikan hasil yang terbaik pada keluaran tiap proses. Skenario pengujian yang digunakan yaitu:

1. Uji coba dengan variasi ukuran pada citra input
2. Uji coba dengan variasi nilai K untuk *k-means clustering*
3. Uji coba dengan variasi skala grid untuk membuat tepian *jaggy*
4. Uji coba kebergunaan

### 5.4 Uji Coba dengan Variasi Ukuran Citra Input

Uji coba dengan variasi ukuran digunakan untuk mengetahui pengaruh ukuran terhadap hasil klastering *k-means* dan citra hasil akhir *pixel art*. Pengujian terhadap ukuran citra input 90 x 90 piksel, 270 x 270 piksel, dan 512 x 512 piksel.

Parameter lain yang digunakan untuk uji coba variasi input ini seperti skala grid adalah sebesar 80 skala dan nilai K sebesar 8.

Hasil dari uji coba variasi ukuran citra input dapat dilihat pada **Tabel 5.4**. Dari hasil uji coba tersebut dapat dilihat bahwa untuk citra ukuran 90 x 90 piksel menghasilkan citra dengan tepian *jaggy* yang kurang terlihat. Citra untuk ukuran 270 x 270 piksel dan 512 x 512 piksel menghasilkan citra *jaggy* yang lebih baik.









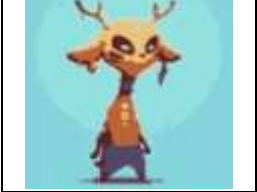
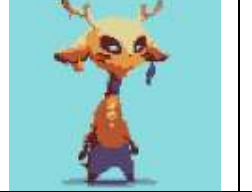







**Tabel 5.3** berikut merupakan tahap pembuatan citra *pixel art* mulai dari input citra asset karakter, pembuatan tepian *jaggy* tanpa segmentasi *k-means*, hingga konversi warna citra *jaggy* menggunakan segmentasi berbasis *k-means*.

**Tabel 5. 3** Tahap pengujian ukuran citra

<b>Citra Input</b>			
<b>Citra Hasil Jaggy, dengan Skala Grid = 100</b>			
<b>Citra Hasil Akhir dengan K-means, nilai K= 8</b>			

Tabel 5. 4 Tabel hasil uji coba variasi ukuran input

Output Citra		
90 x 90	270 x 270	512 x 512
		
		
		
		
		

Output Citra		
90 x 90	270 x 270	512 x 512
		
		
		
		
		

**5.5 Uji Coba dengan Variasi Nilai K untuk K-means Clustering**

Variasi nilai K dilakukan untuk mengetahui pengaruh nilai K tersebut terhadap hasil citra yang dilakukan *clustering* dan output citra *pixel art*. Variasi input nilai K yang digunakan adalah 4, 8, 16, dan 25.

Parameter lain yang digunakan untuk uji coba variasi skala grid ini seperti ukuran citra input adalah ukuran masing-masing citra pada kelompok 3, yaitu 512 x 512 dan skala grid yang digunakan adalah 80.

Hasil uji coba dengan variasi nilai K ditunjukkan pada **Tabel 5.6**. Pada hasil uji coba tersebut dapat disimpulkan bahwa nilai K berpengaruh pada jumlah variasi warna citra hasil. Semakin tinggi nilai K, semakin banyak pula warna citra hasil yang ditampilkan. **Tabel 5.5** berikut merupakan tahap perubahan citra mulai dari input, segmentasi berbasis *k-means*, hingga pembuatan tepian *jaggy*.

























**Tabel 5. 5** Tahap perubahan citra pada pengujian nilai K

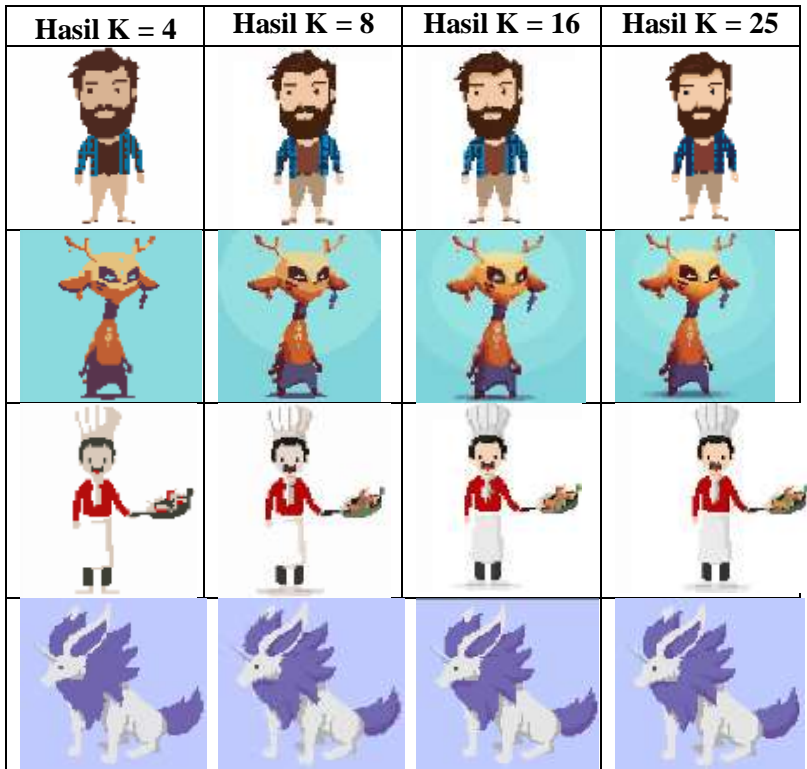
<b>Citra Input</b>	
--------------------	--

<p><b>Citra Hasil K-means dengan nilai K = 4</b></p>	
<p><b>Citra Hasil Akhir dengan Jaggy, nilai Skala Grid = 80</b></p>	

Nilai K kecil berguna untuk membuat warna *gradient* yang terlalu rumit menjadi sederhana, sehingga dapat membuat citra *pixel art* yang baik. Namun, nilai K kecil memiliki kelemahan yaitu, adanya bagian citra yang kurang detail. Pada citra nomor 5 ditunjukkan bahwa dengan K bernilai 4, 8, dan 16 tidak muncul hidung dari citra karakter, sedangkan dengan K bernilai 25 dapat terlihat hidung dari citra karakter tersebut. Selain itu, pada citra nomor 9 juga menunjukkan hasil K = 4 detail makanan yang dipegang oleh citra karakter koki kurang jelas, namun pada nilai K 8, 16, dan 25 terlihat jelas.

**Tabel 5. 6** Tabel hasil uji coba variasi nilai K

Hasil K = 4	Hasil K = 8	Hasil K = 16	Hasil K = 25
			
			
			
			
			
			



## 5.6 Uji Coba dengan Variasi Skala Grid untuk Membuat Tepian Jaggy

Variasi skala grid dilakukan untuk mengetahui pengaruh skala grid terhadap output citra *pixel art*. Variasi input skala yang digunakan adalah 20 (menyatakan skala yang sangat besar), 55 (menyatakan skala yang normal), dan 100 (menyatakan skala yang sangat kecil).

Parameter lain yang digunakan untuk uji coba variasi skala grid ini seperti ukuran citra input adalah ukuran masing-masing citra pada kelompok 3, yaitu 512 x 512 dan nilai K sebesar 8.

**Tabel 5.7** berikut merupakan tahap perubahan citra mulai dari

input citra, segmentasi citra berbasis *k-means clustering*, hingga output citra *jaggy*.

**Tabel 5. 7** Tahap pengujian skala grid tepian










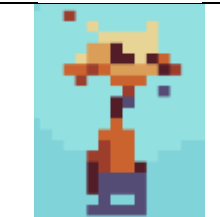

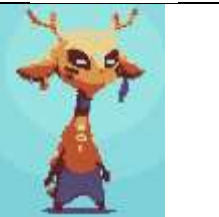



<b>Citra Input</b>	
<b>Citra Hasil Jaggy, dengan Skala Grid = 100</b>	
<b>Citra Hasil Akhir dengan K-means, nilai K= 8</b>	



Hasil uji coba dapat dilihat pada **Tabel 5.8**. Hasil uji coba dengan variasi nilai skala grid menunjukkan bahwa semakin tinggi nilai skala, bentuk kotak-kotak citra hasil akan semakin kecil, dan tepian akan semakin halus atau mendekati bentuk asli.

**Tabel 5. 8** Tabel hasil uji coba variasi nilai skala grid

Hasil skala 20	Hasil skala 55	Hasil skala 100
		
		
		
		

Hasil skala 20	Hasil skala 55	Hasil skala 100
		
		
		
		
		



### 5.7 Uji Coba Kebergunaan

Pengujian kebergunaan ini dilakukan untuk mengetahui tingkat kebergunaan aplikasi terhadap pengguna. Pengujian dilakukan memberikan kesempatan kepada CMO (*Credit Marketing Officer*) dari Maulidan Games, yang bernama Muhammad Hildi Radya Nararya, untuk mencoba aplikasi *Pixel Art Converter*. Setelah pengujian aplikasi *Pixel Art Converter*, pengguna akan diminta untuk mengisi kuesioner untuk mengetahui tanggapan dari pengguna. Rincian pertanyaan dari kuesioner dapat dilihat pada **Tabel 5.9**. Hasil dari kuesioner yang telah diisi oleh CMO Maulidan Games dapat dilihat pada **Tabel 5.10**.

**Tabel 5. 9** Pertanyaan pengujian kebergunaan

No	Pertanyaan	Penilaian
1	Apakah warna citra hasil konversi citra karakter menjadi <i>pixel art</i> sudah sesuai dengan aturan warna pada gaya desain <i>pixel art</i> ?	Sudah/Belum
2	Apakah bentuk kotak-kotak/ <i>jaggy</i> pada hasil hasil konversi citra karakter menjadi <i>pixel art</i> sudah sesuai dengan gaya desain <i>pixel art</i> ?	Sudah/Belum
3	Apakah <i>Pixel Art Converter</i>	Sudah/Belum

No	Pertanyaan	Penilaian
	sudah dapat digunakan untuk citra asset karakter Maulidan Games?	
4	Apakah pilihan input skala warna <i>K-means</i> membantu proses konversi?	Tidak Membantu/ Cukup Membantu/ Sangat Membantu
5	Apakah pilihan input skala Grid Tepian membantu proses konversi?	Tidak Membantu/ Cukup Membantu/ Sangat Membantu
6	Untuk citra <i>gradient</i> berapakah input optimal untuk skala warna yang dibutuhkan?	4/8/16/25
7	Untuk citra <i>flat</i> berapakah input optimal untuk skala warna yang dibutuhkan?	4/8/16/25
8	Berapakah input optimal skala Grid Tepian yang dibutuhkan untuk citra?	20/80/100
9	Berapakah input optimal ukuran citra?	1. Kurang dari 100 x 100 piksel 2. Antara 100 x 100 piksel hingga 200 x 200 piksel 3. Antara 200 x 200 piksel hingga 300 x 300 piksel 4. Antara 300 x 300 piksel hingga 400 x 400 piksel 5. Lebih dari 400 x 400 piksel
10	Tolong berikan saran dan kritik membangun untuk aplikasi <i>Pixel Art Converter</i> untuk dapat dikembangkan lebih jauh.	

**Tabel 5. 10** Hasil kuesioner pengguna

No	Pertanyaan	Penilaian
1	Apakah warna citra hasil konversi citra karakter menjadi <i>pixel art</i> sudah sesuai dengan aturan warna pada gaya desain <i>pixel art</i> ?	Sudah
2	Apakah bentuk kotak-kotak/ <i>jaggy</i> pada hasil konversi citra karakter menjadi <i>pixel art</i> sudah sesuai dengan gaya desain <i>pixel art</i> ?	Sudah
3	Apakah <i>Pixel Art Converter</i> sudah dapat digunakan untuk citra asset karakter Maulidan Games?	Sudah
4	Apakah pilihan input skala warna <i>K-means</i> membantu proses konversi?	Sangat Membantu
5	Apakah pilihan input skala Grid Tepian membantu proses konversi?	Sangat Membantu
6	Untuk citra <i>gradient</i> berapakah input optimal untuk skala warna yang dibutuhkan?	25
7	Untuk citra <i>flat</i> berapakah input optimal untuk skala warna yang dibutuhkan?	16
8	Berapakah input optimal	80

No	Pertanyaan	Penilaian
	skala Grid Tepian yang dibutuhkan untuk citra?	
9	Berapakah input optimal ukuran citra?	Kurang dari 100 x 100 piksel
10	Tolong berikan saran dan kritik membangun untuk aplikasi <i>Pixel Art Converter</i> untuk dapat dikembangkan lebih jauh.	<p>Skala Grid Tepian diberikan pilihan 8, 16, 32, 64, 128, 256. Jadi range Grid Tepian diperpanjang sampai 256.</p> <p>Warna K-means yang 16 secara praktik bagus pada Grid Tepian range 128 dan 256. Tetapi belum disediakan Aplikasi untuk range Grid Tepian tersebut.</p> <p>Pada proses konversi gambar yang lama, ditambahkan loading screen. Sehingga user tahu kalau aplikasi sedang berjalan.</p> <p>Diberikan opsi modifikasi warna RYB/RGB dari citra hasil konversi, sehingga kalau pengguna ingin mengubah warna citra menjadi, misal style Game Boy Advance yang 8 bit, tinggal menekan opsi ini.</p>

## 5.8 Evaluasi

Hasil uji coba *pixel art converter* dengan variasi ukuran input dapat menunjukkan bahwa citra hasil memiliki hasil *pixel art* yang optimal dengan ukuran yang relatif besar. Ukuran yang dapat digunakan yaitu antara 270 x 270 piksel dan 512 x 512 seperti yang ditunjukkan pada uji coba. Citra yang memiliki ukuran kecil, seperti yang ditunjukkan pada uji coba yaitu 90, menghasilkan citra *pixel art* yang kurang baik. Citra ukuran kecil

tersebut, terlihat kabur, kotak-kotak atau *jaggy* yang kurang terlihat, sehingga tidak dapat dimanfaatkan oleh tim desainer Maulidan Games.

Citra dengan jenis desain *gradient color* tidak dapat langsung diubah menjadi bentuk *pixel art* karena jenis warna yang rumit. Warna tersebut harus disederhanakan dengan segmentasi berbasis *k-means clustering*. Hasil segmentasi tersebut dapat diubah menyerupai citra dengan jenis *flat color*. Sehingga, citra *gradient color* yang telah tersegmentasi tersebut dapat mendapat hasil *pixel art* yang baik. Berikut contoh perbandingan citra *gradient color* yang di-segmentasi terlebih dahulu sebelum menjadi *pixel art* dan citra tanpa segmentasi.



**Gambar 5.1** Perbandingan citra dengan segmentasi dan tanpa segmentasi (a) tanpa segmentasi (b) dengan segmentasi

Citra pada **Gambar 5.1 (a)** menunjukkan bahwa tanpa dilakukan segmentasi berbasis *k-means clustering*, citra hasil *pixel art* memuat warna *gradient* yang rumit dan bagian mulut citra karakter terlihat rusak. Berbeda dengan hasil segmentasi berbasis *k-means clustering* pada **Gambar 5.1 (b)** memiliki hasil warna lebih sederhana seperti *flat color* dan bagian mulut citra karakter terlihat lebih rapi.

*Pixel art* memiliki ciri khas bentuk tepian bergerigi/*jaggy*. Pada *pixel art converter* yang dibangun dipengaruhi oleh input skala grid oleh pengguna. Pada uji coba yang dilakukan dengan skenario variasi input skala grid, dapat ditunjukkan bahwa semakin tinggi nilai skala, hasil *pixel art* semakin halus atau menyerupai bentuk asli.

Berdasarkan pengujian aplikasi *Pixel Art Converter* oleh tim desain dari Maulidan Games yaitu Muhammad Hildi R. N. yaitu selaku *Credit Marketing Officer*, aplikasi ini dinyatakan telah dapat digunakan oleh tim desain Maulidan Games. Input dan proses, serta hasil output yang didapat telah sesuai dengan kebutuhan yang dibutuhkan. Output citra *pixel art* sudah sesuai dengan gaya *pixel art* yang digunakan oleh para *designer*.

Terdapat beberapa nilai optimal yang dapat digunakan oleh tim desain Maulidan Games supaya mendapatkan hasil citra *pixel art* yang baik. Berikut merupakan nilai-nilai optimal yang diberikan oleh CMO Maulidan Games,

1. Untuk citra *gradient color* nilai optimal untuk skala warna atau nilai K untuk *K-means* adalah 25, sedangkan untuk citra *flat color* menggunakan nilai K 16.
2. Nilai skala grid untuk membuat tepian adalah 80.
3. Ukuran citra yang baik untuk dilakukan konversi menjadi *pixel art* adalah 100 x 100 piksel.

Namun, perlu adanya pengembangan, yaitu:

1. Memberikan *range* skala grid dengan pilihan: 8, 16, 32, 64, 128, 256.
2. Diberikan indikator (seperti *loading screen*) yang menyatakan sedang dilakukan proses konversi.
3. Memberikan pilihan modifikasi warna RYB/RGB dari citra hasil konversi, sehingga apabila pengguna dapat mengubah menjadi *style* warna 8 bit.



## **BAB VI**

### **KESIMPULAN DAN SARAN**

Bab ini membahas mengenai kesimpulan yang dapat diambil dari hasil uji coba *pixel art converter* yang telah dilakukan sebagai jawaban dari rumusan masalah. Selain itu juga terdapat saran yang ditujukan untuk pengembangan penelitian lebih lanjut.

#### **6.1 Kesimpulan**

Berdasarkan tugas akhir ini dapat ditarik kesimpulan bahwa:

1. Rancang bangun *pixel art converter* dapat dibangun dengan segmentasi berdasarkan *k-means clustering* dengan cara memberikan input citra dan nilai K yang akan mempengaruhi warna citra *pixel art* yang dihasilkan.
2. Nilai skala grid untuk membuat tepian *jaggy* mempengaruhi hasil citra *pixel art*. Semakin besar nilai skala grid, citra hasil *pixel art* akan semakin halus atau mendekati bentuk asli citra inputnya dan semakin kecil nilai skala grid, bentuk kotak-kotak (tepi *jaggy*) akan semakin besar dan menjauhi bentuk asli dari citra input.
3. Nilai K untuk segmentasi berbasis *k-means clustering* mempengaruhi warna citra hasil *pixel art*. Semakin tinggi nilai K, maka jumlah variasi warna pada citra hasil *pixel art* semakin banyak.
4. Segmentasi berbasis *k-means clustering* dapat mengubah citra dengan jenis *gradient color* menyerupai *flat color* sehingga dapat memberikan citra hasil *pixel art* yang lebih baik.
5. Ukuran citra mempengaruhi kualitas citra hasil *pixel art*. Semakin tinggi ukuran citra input, kualitas citra hasil *pixel art* juga akan semakin baik.

6. Menurut CMO Maulidan Games, aplikasi *Pixel Art Converter* telah memenuhi kebutuhan industri dan dapat digunakan oleh tim desain Maulidan Games.
7. Menurut CMO Maulidan Games, citra hasil *pixel art* yang optimal didapatkan dari input:
  - a. Untuk citra *gradient color* nilai optimal untuk skala warna atau nilai K untuk *K-means* adalah 25, sedangkan untuk citra *flat color* menggunakan nilai K 16.
  - b. Nilai skala grid untuk membuat tepian adalah 80.
  - c. Ukuran citra yang baik untuk dilakukan konversi menjadi piksel art adalah 100 x 100 piksel.

## 6.2 Saran

Berikut saran yang diberikan untuk tugas akhir ini:













1. Dapat dikembangkan fitur-fitur lain, seperti menyimpan gambar dan menghilangkan background citra karakter
2. Perlu adanya analisa parameter lain yang dapat digunakan untuk memperbaiki *pixel art converter*
3. Memberikan *range* skala grid dengan pilihan: 8, 16, 32, 64, 128, 256.
4. Diberikan indikator (seperti *loading screen*) yang menyatakan sedang dilakukan proses konversi.
5. Memberikan pilihan modifikasi warna RYB/RGB dari citra hasil konversi, sehingga apabila pengguna dapat mengubah menjadi *style* warna 8 bit.







## LAMPIRAN

### B. Tahap Uji Coba dengan Variasi Skala Grid untuk Tepian Jaggy

#### 1. Skala 20













Citra Input	Citra Hasil Jaggy	Citra Hasil Akhir dengan K-means
		
		
		
		













Citra Input	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir dengan <i>K-means</i>
		
		
		
		

Citra Input	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir dengan <i>K-means</i>
		
		













## 2. Skala 55

Citra Input	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir dengan <i>K-means</i>
		
		




Citra Input	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir dengan <i>K-means</i>
		
		
		
		





Citra Input	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir dengan <i>K-means</i>
		
		
		
		

3. Skala 100

Citra Input	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir dengan <i>K-means</i>
		
		
		
		


















Citra Input	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir dengan <i>K-means</i>
		
		
		
		



Citra Input	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir dengan <i>K-means</i>
		
		

C. Tahap Uji Coba dengan Variasi Nilai K

1. Nilai K = 4

Citra Input	Citra Hasil <i>K-means</i>	Citra Hasil Akhir dengan <i>Jaggy</i>
		
		

Citra Input	Citra Hasil K-means	Citra Hasil Akhir dengan Jaggy
		
		
		
		
		

Citra Input	Citra Hasil K-means	Citra Hasil Akhir dengan Jaggy
		
		
		

2. Nilai K = 8

Citra Input	Citra Hasil K-means	Citra Hasil Akhir dengan Jaggy
		

Citra Input	Citra Hasil K-means	Citra Hasil Akhir dengan Jaggy
		
		
		
		
		

Citra Input	Citra Hasil K-means	Citra Hasil Akhir dengan Jaggy
		
		
		
		

### 3. Nilai $K = 16$

Citra Input	Citra Hasil K-means	Citra Hasil Akhir dengan Jaggy
		
		
		
		
		



Citra Input	Citra Hasil K-means	Citra Hasil Akhir dengan Jaggy
		
		
		
		
		



#### 4. Nilai $K = 25$








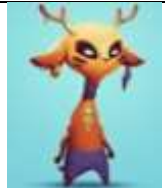
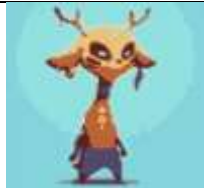






Citra Input	Citra Hasil K-means	Citra Hasil Akhir dengan Jaggy
		
		
		
		
		

Citra Input	Citra Hasil K-means	Citra Hasil Akhir dengan Jaggy
		
		
		
		
		

#### D. Tahap Uji Coba dengan Variasi Ukuran Citra













##### 1. Ukuran 90 x 90 piksel

Citra Hasil K-means	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir
		
		
		
		
		

Citra Hasil K- means	Citra Hasil Jaggy	Citra Hasil Akhir
		
		
		
		
		

## 2. Ukuran 270 x 270 piksel

Citra Hasil K-means	Citra Hasil Jaggy	Citra Hasil Akhir
		
		
		
		
		







Citra Hasil K- <i>means</i>	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir
		
		
		
		

Citra Hasil K-means	Citra Hasil Jaggy	Citra Hasil Akhir
		

### 3. Ukuran 512 x 512 piksel

Citra Hasil K-means	Citra Hasil Jaggy	Citra Hasil Akhir
		
		
		



Citra Hasil K-means	Citra Hasil <i>Jaggy</i>	Citra Hasil Akhir
		
		
		
		
		



Citra Hasil K-means	Citra Hasil Jaggy	Citra Hasil Akhir
		
		

## E. Hasil Kuesioner Pengujian Kebergunaan

PERTANYAAN      TANGGAPAN      1

---

Tanggapan tidak dapat dihapus

### Form Penilaian Pixel Art Converter

Bagian ini menanggapi tentang kebutuhan sistem

\* Wajib

Apakah warna citra hasil konversi citra karakter menjadi pixel art sudah sesuai dengan aturan warna pada gaya desain pixel art? \*

☒ Sudah

☐ Belum

Apakah bentuk kotak-kotak/jaggy pada hasil hasil konversi citra karakter menjadi pixel art sudah sesuai dengan gaya desain pixel art? \*

☒ Sudah

☐ Belum

**Lampiran 1.** Hasil pengujian kebutuhan pengguna

### Uji Coba oleh Tim Desain

Apakah Pixel Art Converter sudah dapat digunakan untuk citra asset karakter Maulidan Games? \*

- ☒ Sudah
- ☐ Belum

Apakah pilihan input skala warna K-means membantu proses konversi? \*

- ☐ Tidak Membantu
- ☐ Cukup Membantu
- ☒ Sangat Membantu

Apakah pilihan input skala Grid Tepian membantu proses konversi? \*

- ☐ Tidak Membantu
- ☐ Cukup Membantu
- ☒ Sangat Membantu

Untuk citra gradient berapakah input optimal untuk skala warna yang dibutuhkan? \*

25

Untuk citra flat berapakah input optimal untuk skala warna yang dibutuhkan? \*

16

Berapakah input optimal skala Grid Tepian yang dibutuhkan untuk citra? \*

30

Berapakah input optimal ukuran yang dibutuhkan untuk citra? \*

Kurang dari 100 x 100 piksel

### Lampiran 2. Hasil uji coba pengguna

### Saran Membangun untuk Aplikasi Pixel Art Converter

Tolong berikan saran dan kritik membangun untuk aplikasi Pixel Art Converter untuk dapat dikembangkan lebih jauh. \*

Skala Grid Tepian diberikan pilihan 8, 16, 32, 64, 128, 256. Jadi range Grid Tepian diperpanjang sampai 256.

Warna K-means yang 16 secara praktik bagus pada Grid Tepian range 128 dan 256. Tetapi belum disediakan Aplikasi untuk range Grid Tepian tersebut.

Pada proses konversi gambar yang lama, ditambahkan loading screen. Sehingga user tahu kalau aplikasi sedang berjalan.

Diberikan opsi modifikasi warna RYB/RGB dari citra hasil konversi, sehingga kalau pengguna ingin mengubah warna citra menjadi, misal style Game Boy Advance yang 8 bit, tinggal menekan opsi ini.

---

### Lampiran 3. Saran Pengguna

*[Halaman ini sengaja dikosongkan]*

## DAFTAR PUSTAKA

- [1] R. Prayuga, "Apakah itu Pixel Art?," 16 Februari 2015. [Online]. Available: <http://inovasipintar.com/apakah-itu-pixel-art/>. [Accessed 20 Oktober 2016].
- [2] Wadezig, "8-Bit Bagian Ketiga: Grafis dan Pixel Art," 12 Mei 2014. [Online]. Available: <http://www.wadezig.com/8-bit-bagian-ketiga-grafis-dan-pixel-art/>. [Accessed 20 Oktober 2016].
- [3] RedBubble, "Redbubble," Redbubble, 20 July 2016. [Online]. Available: <https://www.redbubble.com/people/gustavinlavin/works/14766861-super-mario-bros-pixel?p=art-print>. [Accessed 16 July 2017].
- [4] Steam, "Steam," Steam, 8 Agustus 2016. [Online]. Available: [http://store.steampowered.com/app/269310/Infectorator\\_\\_Survivors/](http://store.steampowered.com/app/269310/Infectorator__Survivors/). [Accessed 12 November 2016].
- [5] Pixel Stitch, "Pixel Stitch," Pixel Stitch, Agustus 2013. [Online]. Available: <http://www.pixel-stitch.net/index.html>. [Accessed 12 November 2016].
- [6] Luna Pic, "Luna Pic," Luna Pic, 28 Maret 2015. [Online]. Available: <http://www194.lunapic.com>. [Accessed 12 November 2016].
- [7] I. Septyan, "Rekonstruksi Citra Menggunakan Algoritma Structure-Adaptive Normalized Convolution," *Rekonstruksi Citra Menggunakan Algoritma Structure-Adaptive Normalized Convolution*, p. 2, 2011.
- [8] E. R. Swedia and M. Cahyanti, Algoritma Transformasi Ruang Warna, Depok, 2010.
- [9] R. Kurnia and S. Nurhadi, "Deteksi Obyek Berbasis Warna dan Ukuran dengan Bantuan Interaksi Komputer-Manusia," *Deteksi Obyek Berbasis Warna dan Ukuran dengan Bantuan Interaksi Komputer-Manusia*, p. 11, 2008.
- [10] G. Shir, "Muzk," 12 April 2017. [Online]. Available: <https://medium.muz.li/why-gradients-are-the-new-colors->

3d8d42a7a6fc. [Accessed 17 Juli 2017].

- [11] P. Hill, Artist, *Littlest Pet Shop character illustration*. [Art]. Hasbro, Inc.
- [12] I. Burmistrov, T. Zlokazova, A. Izmalkova and A. Leonova, "Flat Design vs Traditional Design: Comparative Experimental Study," *Flat Design vs Traditional Design: Comparative Experimental Study*, pp. 107 - 108.
- [13] Flaticon, Artist, *Animal Set*. [Art]. Freepik.
- [14] A. Z. Arifin, A. Yuniarti and L. R. Dewi, "Segmentasi Trabecular Bone berdasarkan Linear Structure pada Citra Dental Panoramic Radiograph," *POMITS*, 2009.
- [15] S. Widodo, A. Hidayanto and R. R. Isnanto, "Segmentasi Citra Menggunakan Teknik Pemetaan Warna (Color Mapping) dengan Bahasa Pemrograman Delphi," *Segmentasi Citra Menggunakan Teknik Pemetaan Warna (Color Mapping) dengan Bahasa Pemrograman Delphi*, p. 2, 2016.
- [16] Y.-H. Wang, "Tutorial: Image Segmentation," *Tutorial: Image Segmentation*, p. 21, 2010.
- [17] S. Agustina, D. Yhudo, H. Santoso, N. Marnasusanto, A. Tirtana and F. Khusnu, "Clustering Kualitas Beras Berdasarkan Ciri Fisik Menggunakan Metode K-means," *Clustering Kualitas Beras Berdasarkan Ciri Fisik Menggunakan Metode K-means*, p. 2.



## BIODATA PENULIS

Yuna Sugianela, dilahirkan di Kediri pada tanggal 27 Juni 1995. Penulis telah menyelesaikan pendidikan formal di TK Dharmawanita Sambirobyong, SD Negeri Menang, SMPN 1 Ngasem, dan SMAN 2 Kediri. Setelah lulus SMA pada tahun 2013, penulis diterima di jurusan Teknik Informatika ITS melalui jalur SNMPTN. Penulis pernah menjadi asisten dosen pada mata kuliah Sistem Digital, Sistem Teknologi Informasi, dan Manajemen Basis Data. Selain itu, pernah menjadi asisten dosen PIKTI tahun 2016 dan 2017 Di Teknik Informatika Penulis mengambil rumpun mata kuliah Komputasi Cerdas dan Visi dan berkesempatan menjadi Administrator laboratorium tersebut pada tahun 2015 hingga 2016.

Penulis aktif pada organisasi selama perkuliahan. Pada tahun kedua perkuliahan penulis mengikuti organisasi Himpunan Mahasiswa Teknik Computer-Informatika sebagai staff Pengembangan Sumber Daya Mahasiswa, Badan Eksekutif Mahasiswa Fakultas Teknologi Informasi sebagai staff *Research and Technology Department*, Keluarga Muslim Informatika (KMI) sebagai staff kaderisasi. Pada tahun ketiga perkuliahan, penulis melanjutkan organisasi KMI sebagai kepala divisi Al-Afifah dan Badan Eksekutif Mahasiswa ITS (BEM ITS) sebagai Asisten Menteri Keuangan. Pada tahun terakhir penulis masih aktif berorganisasi di BEM ITS sebagai Bendahara Eksekutif.

Selain organisasi penulis juga aktif pada kegiatan kampus seperti ITS Expo sejak 2014 hingga 2016 dan Schematics 2014, pada bidang minat bakat yaitu UKM Fotografi, dan aktif pada perlombaan seperti Gemastik dan PKM.

*[Halaman ini sengaja dikosongkan]*